

## **REMARKS**

### **INTRODUCTION**

Claims 25-29 and 34-37 were previously and are currently pending and under consideration.

Claims 25-29 and 34-37 are rejected.

No new matter is being presented, and approval and entry are respectfully requested.

### **ENTRY OF AMENDMENT UNDER 37 CFR §1.116**

Applicant requests entry of this Rule 116 Response because no amendments alter the scope of the claims and no new features or new issues are being raised.

The Manual of Patent Examining Procedures sets forth in Section 714.12 that "any amendment that would place the case either in condition for allowance or in better form for appeal may be entered." Moreover, Section 714.13 sets forth that "the Proposed Amendment should be given sufficient consideration to determine whether the claims are in condition for allowance and/or whether the issues on appeal are simplified." The Manual of Patent Examining Procedures further articulates that the reason for any non-entry should be explained expressly in the Advisory Action.

### **REJECTIONS UNDER 35 USC § 103**

In the Office Action, at pages 2-4, claims 25-29 and 34-37 were rejected under 35 U.S.C. § 103 as being unpatentable over Judge in view of Official Notice and further in view of Myers. This rejection is traversed and reconsideration is requested.

### **PROPOSED MODIFICATION WOULD SUBSTANTIALLY ALTER JUDGE AND WOULD RENDER IT UNFIT FOR ITS INTENDED PURPOSE**

MPEP § 2143.01 (end of section), states that there is no prima facie case of obviousness

when the suggested combination of references would require a substantial reconstruction and redesign of the elements shown in the primary reference as well as a change in the basic principle under which the primary reference construction was designed to operate.

MPEP § 2143.01 also states that if a proposed modification would render the prior art being modified unsatisfactory for its intended purpose then there is no suggestion or motivation to make the proposed modification.

Applicant will show below how the proposed modification of Judge would require substantial reconstruction and would render it unfit for its intended purpose. In general, this will be shown by summarizing the relevant claim feature, reviewing the teachings of Judge and its functionality, restating the proposed modification of Judge, expanding on the suggested motive for the modification, and explaining what would be involved in extending the functionality of Judge to a server.

#### Claimed DMS Server

Claim 1, for example, recites a document management server. More specifically, claim 1 recites using an Application Programming Interface (API) to enable a healthcare system (HCS) to access a document management system (DMS) server managing patient documents and making them available for access by plural user terminals. The DMS server is clearly a network-type server in that receives user requests, manages documents and makes them available to user terminals (clients), authorizes access, etc. Furthermore, document servers are well known in the art as network servers that concurrently serve documents over a network to plural users/terminals.

#### Teachings of Judge, in Particular Functionality of Judge

Judge is a system for a single user running different applications on a workstation. Judge allows information to be shared between these applications. If a user on a workstation switches from one application to another, shared memory on the workstation is used to exchange patient context information between the two applications. The patient information is "data about the same patient" (Abstract). For example, shared patient data may be information such as a patient's medical record number, a current time, a department, etc. In other words, if a clinician is using an application to view data about a specific disease of a patient, when the clinician switches to another application, the other application receives, via shared memory, the

disease currently active in the first application. The overriding purpose of Judge is to allow a user "to review data about a single patient using more than one program" without having to continually reenter the same data about the patient (column 1, lines 22-24).

The mechanism for inter-application information sharing in Judge is shared memory. As discussed further below, shared memory is available only on a local workstation. Furthermore, shared memory is only a conduit for information. To make the exchange of information between different applications possible, Judge provides an application programming interface (API) which any application may be programmed with. The API abstracts the shared information such as a patient ID, disease, etc. The API also provides a convenient interface to access the shared memory.

The local interprocess or inter-application nature of Judge is also apparent from how it is used. Judge teaches that different healthcare applications may be executing on the same workstation. As is well known in graphical interfaces, it is possible to switch between different applications running on a workstation, where the current application is the focus application. Judge explains in lengthy detail a solution for assuring that switching focus between different applications triggers an exchange of patient context information between those applications. Judge explicitly states that the "PCI embodiment provides a mechanism for changing the system focus from one running application to another running application" (column 17, lines 39-42).

In sum, the function of Judge is to allow a single specific workstation user to automatically share the context of their activity (e.g. working on a particular disease or a particular patient) when switching between different applications on a workstation.

#### Proposed Modification of Judge

At page 3, the rejection states that Judge "does not disclose a server system or multiple computers". The rejection states that Myers teaches that a medical information system can be distributed into document servers that supply documents to multiple computer workstations. The modification is described as modifying Judge "so as to apply the API and new user interfaces to document server computers (10, 12, 14) instead of just an individual computer and distribute the data to multiple workstations".

Applicant understands this to mean that the patient context information sharing of Judge

is proposed to be modified to function as a document server.

Function of Judge and Suggested Motive for Modification of Judge

The rejection states that the proposed modification "would afford the added functionalities created by the API to be usable across multiple computer systems and a large number of users".

Applicant understands that the suggested motive is that it would have been desirable to make the functionalities of Judge available or usable to multiple concurrent users. In other words, it would have been somehow beneficial for different users on different workstations to know what patient, disease, etc. was the subject/target of another user using an application on another computer.

Modifying Judge to be a Server is Impractical and Would Require Complete Redesign of Judge

Judge discloses a system in which applications on a same workstation share information about the context (particular patient) in which they are being used. So, if a doctor is using a chart application for patient X, and the doctor switches to a pharmacy application, the pharmacy application can automatically orient to patient X; the doctor does not need to tell the pharmacy application to access patient X – this happens automatically.

As discussed above, the mechanism that Judge uses is shared memory. However, shared memory is not practical for implementing a server or for sharing information over a network as a server would. It is well known in the art of computer programming that shared memory is used for communication between processes executing on the same machine under the same operating system instance.

As stated in The Code Project, shared memory can only be used for a client/server systems where the clients and server exist only locally on one machine. ("Since shared memory may be used only on a single machine, all processes must run on a same computer. This is an obvious limitation. However, the advantage is a speed of communication (note that speed may be severely compromised by a bad implementation on both sides - client and server ... )", <http://www.thecodeproject.com/threads/sharedmemipc.asp>, copy included herewith), emphasis added.

Also, as stated at Developer.com, a problem with shared memory is that "[a]ccess to shared memory must be protected from concurrent access which results in data corruption – synchronization". Furthermore, "[shared memory] is a very powerful mechanism to transfer large amount of data between the client and the server. It is also very fast (actual interprocess communication) and secure. However, it is limited to the local machine - it does not work via a network" (emphasis added, <http://www.developer.com/net/cplus/article.php/632001>, copy included).

It is well known in the art that shared memory is not adaptable to a multi-user network server because of the difficulty of providing protection for concurrent access, synchronization, and avoiding corruption of data. In Judge, only one application has focus at one time, and multiple applications do not concurrently access the shared memory at one time; concurrency is not an issue. Rather, the shared memory is accessed in turn when there is a switch to another application. The proposed modification of Judge would require discarding the shared memory design and redesigning it from the ground up. A prima facie case of obviousness has not been made because this kind of substantial redesign and reconstruction has been explicitly held to be non-obvious.

Withdrawal of the rejection is respectfully requested.

A prima facie case of obviousness has also not been made because the primary mode of operation of Judge would be changed. The primary mode of operation in Judge is single user operation. As discussed above, Judge is designed for the purpose of allowing a single user to share information between two applications that the user is using on a workstation. Its mode of operation is intrinsically single-user. Multi-user document serving is a fundamentally different mode of operation.

Judge cites Hewlett Packard's CareVue product as an exemplary system in which it can be used. However, a review of the CareVue product shows that it is itself a single-vendor client/server system. Clients or workstations (such as those in Judge) use multiple applications to access backend database servers, preferably using ODBC. In other words, the applications in Judge are clients that access database servers (e.g. CareVue database servers). It would not make sense to modify an inter-application helper interface (Judge) to operate as a server, particularly when database servers already handle data sharing between computers/users.

Withdrawal of the rejection is respectfully requested.

Proposed Modification Would Render Judge Unfit For Its Intended Purpose

The purpose of Judge is to allow a user to transfer information between applications *when switching focus from one application to another* (possibly by launching a new application, or interactively selecting another application). Shared memory is used because the functionality need only be local and fast. Modifying Judge to be a server would require an access to a server every time a user switched applications. Even short delays for accessing a server would significantly impair the process of locally changing between applications. Judge et al. chose shared memory because their intended purpose was purely local. Modifying Judge to operate as a server would render it unsatisfactory for its intended purpose of sharing information when local applications change focus. Therefore, there is no suggestion or motivation to make the proposed modification.

Withdrawal of the rejection is respectfully requested.

In sum, Judge is for a user to automatically share information between the applications that they are locally using. The rejection proposes that making this available for a document server "would afford the added functionalities created by [Judge's shared memory] API to be usable across multiple computer systems and a large number of users". However Judge's functionality – inter-application communication for a single local user – is impracticable for a server and irrelevant to multiple users. There is no reason for one user to share with another user information about which patient they are currently accessing, which disease they are currently researching, etc. One user does not care which patient another user is working on. Substantive data sharing is handled by a backend database server, as is the case with most prior art healthcare systems.

CLAIMS 25 AND 29: JUDGE LACKS AUTOMATIC LOGON

Claim 25 recites "interactively logging onto the modified HCS by executing an HCS user authentication process of the HCS which is separate from the API", and "based on authentication information related to the HCS user authentication process, automatically authenticating a user to use the DMS server".

Claim 29 recites that "the interactive logging onto the modified HCS comprises a healthcare worker providing log-on information to the modified HCS using the user interface of the HCS, and wherein the automatically authenticating to the DMS comprises the API automatically logging the user to the document management system using the log-on information from the user interface".

In other words, an existing Healthcare System can be modified such that the user can use one interface to log in and have user validation for two systems. This is not disclosed or suggested in Judge, which deals only with local applications on one workstation.

The rejection notes that "the API performs a logging operation where applications are registered into the PCI. This is also an authentication process since the PCI can issue responses that applications being loaded are successful or invalid". Registering an application with a local service (the API) does not require user authentication. In fact, the purpose of Judge is to allow sharing of Patient Context Information, not authentication information. Columns 17 and 18 in Judge discuss the Switch-to button as merely switching control to the other application. There is no discussion or suggestion of user authentication when switching to those applications.

Furthermore, as shown above, Judge cannot be modified into a server. Therefore, there is not a prima facie case of automatic authentication to a DMS server. The Official Notice only establishes that login authentication is known art. However, automatic authentication based on application switching could present serious problems in the privacy-sensitive healthcare context. Withdrawal of the rejection is further respectfully requested.

## **CONCLUSION**

There being no further outstanding objections or rejections, it is submitted that the application is in condition for allowance. An early action to that effect is courteously solicited.


Finally, if there are any formal matters remaining after this response, the Examiner is requested to telephone the undersigned to attend to these matters.

If there are any additional fees associated with filing of this Amendment, please charge the same to our Deposit Account No. 19-3935.

Respectfully submitted,

STAAS & HALSEY LLP

Date: Nov 1, 2009

By:   
James T. Strom  
Registration No. 48,702

1201 New York Ave, N.W., Suite 700  
Washington, D.C. 20005  
Telephone: (202) 434-1500  
Facsimile: (202) 434-1501